Learning Icosahedral Spherical Probability Map Based on Bingham Mixture Model for Vanishing Point Estimation

Haoang Li1,* Kai Chen1,* Pyojin Kim2 Kuk-Jin Yoon3 Zhe Liu4 Kyungdon Joo5,† Yun-Hui Liu1,†
1The Chinese University of Hong Kong, Hong Kong, China 2Sookmyung Women’s University, South Korea 3KAIST, South Korea 4University of Cambridge, United Kingdom 5UNIST, South Korea

Abstract

Existing vanishing point (VP) estimation methods rely on pre-extracted image lines and/or prior knowledge of the number of VPs. However, in practice, this information may be insufficient or unavailable. To solve this problem, we propose a network that treats a perspective image as input and predicts a spherical probability map of VP. Based on this map, we can detect all the VPs. Our method is reliable thanks to four technical novelties. First, we leverage the icosahedral spherical representation to express our probability map. This representation provides uniform pixel distribution, and thus facilitates estimating arbitrary positions of VPs. Second, we design a loss function that enforces the antipodal symmetry and sparsity of our spherical probability map to prevent over-fitting. Third, we generate the ground truth probability map that reasonably expresses the locations and uncertainties of VPs. This map unnecessarily peaks at noisy annotated VPs, and also exhibits various anisotropic dispersions. Fourth, given a predicted probability map, we detect VPs by fitting a Bingham mixture model. This strategy can robustly handle close VPs and provide the confidence level of VP useful for practical applications. Experiments showed that our method achieves the best compromise between generality, accuracy, and efficiency, compared with state-of-the-art approaches.

1. Introduction

Vanishing point (VP) is the intersection of two image lines whose corresponding 3D lines are parallel. VP has various applications such as scene understanding [12], camera orientation estimation [14] and 3D reconstruction [13]. While VP estimation has been widely studied, existing approaches have two main limitations. First, numerous methods [25, 24, 22, 33, 1, 18] rely on pre-extracted image lines, but they are sensitive to the number and quality of lines. For example, given a small number of lines, a method may neglect some VPs (see Figs. 1(a) and 1(b)). Moreover, given several lines corrupted by outliers, e.g., shadow boundaries, a method may mistakenly detect VPs (see Figs. 1(a) and 1(c)). Second, many methods [4, 39, 25, 24, 41] rely on prior knowledge of the number of VPs. They typically assume three orthogonal VPs in Manhattan world [9], and thus neglect partial VPs or result in redundant detection in non-Manhattan scenes [30, 38] (see Figs. 1(a) and 1(d)). While recent approaches [15, 22, 23, 18] can automatically determine the number of VPs, they rely on image lines.

To overcome the above limitations, we propose the first VP estimation method that is independent of image lines and also can automatically determine the number of VPs. Specifically, as shown in Fig. 1(e), the connection between a VP and camera center is aligned to a dominant direction (DD). Compared with VP that may be extremely far from
the image center, unit DD starting at the camera center is enclosed by a unit sphere. We thus follow [41, 22] to reformulate VP estimation as DD computation, i.e., we aim to determine which positions on the sphere correspond to DDs. To achieve this goal, we propose a network that treats a perspective image as input and predicts a spherical probability map of DD. Based on this map, we can detect all the DDs, regardless of the number of DDs.

Our method is reliable thanks to four technical novelties. First, as shown in Fig. 1(f), we leverage the icosahedral spherical representation [2] to express our probability map. This representation provides a more uniform pixel distribution than the widely-used equi-angular discretization on the sphere (see Fig. 2(a)). Accordingly, it facilitates estimating arbitrary orientations of DDs. Second, we design a loss function that not only is effective in fitting data, but also enforces the antipodal symmetry and sparsity of our spherical probability map for regularization. Third, we generate the ground truth map that reasonably expresses the locations and uncertainties of DDs. This map unnecessarily peaks at noisy annotated DDs and also exhibits various anisotropic dispersions (see Fig. 6(d)). We train our network by minimizing the difference between the predicted and ground truth probability maps. Fourth, given a predicted probability map, we detect DDs by fitting a Bingham mixture model [6] (see Fig. 1(f)). This strategy is free of threshold, and thus can handle close DDs more robustly than non-maximum suppression [27]. Moreover, it can provide the confidence level of DD useful for practical applications. Our main contributions are:

- Our method is independent of image lines and also can automatically determine the number of DDs.
- We leverage the icosahedral spherical representation to express our probability map. This representation facilitates estimating arbitrary orientations of DDs.
- We design a loss function that enforces the antipodal symmetry and sparsity of our spherical probability map to prevent over-fitting.
- We introduce a strategy to generate the ground truth probability map that reasonably expresses the locations and uncertainties of DDs.
- We detect DDs by fitting a Bingham mixture model on a predicted map. This strategy is free of threshold and can provide the confidence level of DD.

2. Related Work

We classify existing VP estimation methods into two categories, i.e., traditional and deep learning-based ones.

1To the best of our knowledge, we first introduce icosahedral spherical representation to VP estimation problem. Our work may inspire the community to apply this representation to the other geometric problems, e.g., camera pose estimation.

Traditional Methods. Most traditional methods rely on pre-extracted image lines, i.e., they cluster these lines by unknown-but-sought VPs. Representative approaches [28, 3, 39, 33, 4, 5, 25, 24] assume three orthogonal VPs in Manhattan world. Among them, the sampling-based methods [3, 39] hypothesize several candidate VP triplets and select the optimal one that maximizes the number of inlier lines. They lead to unsatisfactory accuracy since some sampled lines may be affected by noise. The search-based methods [4, 5] search in a parameter space related to rotation and find the optimal parameters that maximize the number of inlier lines. They are accurate but inefficient due to numerous rounds of space sub-division and time-consuming bound computation. A method hybridizing the sampling and search [25] achieves a balance between accuracy and efficiency. Due to the assumption of three VPs, the above methods are prone to neglecting partial VPs or resulting in redundant detection in non-Manhattan scenes. In contrast, recent approaches [15, 22, 23] can automatically determine the number of VPs. However, they lead to unsatisfactory efficiency since their parameter spaces are high-dimensional and their cost functions are highly non-linear.

Deep Learning-based Methods. Earlier method [38] requires pre-extracted image lines. It first uses a network to predict several candidate horizons, and then finds the optimal VPs that maximize the number of inlier lines. Some approaches that do not rely on image lines are proposed [8, 41]. They directly treat an image as input. For example, Chang et al. [8] formulated VP estimation as a classification problem. However, this method can only detect VPs within the image. Zhou et al. [41] adopted a coarse-to-fine strategy to sample points on the sphere. For each sampled point, they used a network to predict the probability of VP. Then they selected the points with top $K$ probabilities ($K$ is the number of VPs). While this method can handle VPs outside the image, it is sensitive to the sampling resolution and also requires prior knowledge of the number of VPs. In addition, an approach for multi-model fitting [18] can automatically determine the number of VPs. However, it requires image lines as input.

3. Predicting Spherical Probability Map

Given a perspective image, our network predicts a spherical probability map of DD. In Section 3.1, we introduce the icosahedral spherical representation that makes our network reliably handle arbitrary orientations of DDs. In Section 3.2, we introduce our network architecture. In Section 3.3, we design a novel loss function that exploits the characteristic of spherical map for regularization.

3.1. Icosahedral Spherical Representation

As shown in Fig. 2(a), the widely-used equi-angular discretization on the sphere [15, 22] results in non-uniform
Accordingly, our experiments show that a baseline method using this representation cannot handle arbitrary orientations of DDs, especially DDs near two poles (see Fig. 8(a)). To solve this problem, we propose to use a novel icosahedral spherical representation [2]. As shown in Fig. 2(b), an icosahedron consists of 20 basic faces with the same area. We sub-divide each face into 4 sub-faces. After $N$ rounds of sub-division, we obtain $20 \times 4^N$ sub-faces. We empirically set $N$ as 5, which leads to reliable DD estimation in our experiments. As shown in Fig. 2(c), we extrude all the vertices of icosahedron sub-faces to the unit sphere, obtaining the icosahedral spherical representation. We use this representation to express our spherical probability map. Specifically, a sub-face of this representation defines a pixel of spherical map. We associate each pixel with the probability that a DD passes through this pixel (see Fig. 1(f)).

In addition, as shown in Fig. 1(f), a DD $d$ is up to sign, i.e., $d$ and $-d$ are equivalent in DD estimation [32]. Accordingly, antipodal pixels of our spherical map should be associated with the same probability of DD. Despite this antipodal symmetry, we do not use the hemisphere to express the probability map. The reason is that a dividing line of sphere, i.e., a great circle (see Fig. 9(a)) may split a probability distribution on the map and further affects the network training, as will be shown in the experiments.

### 3.2. Network Architecture

As shown in Fig. 3(a), our network is based on encoder-decoder architecture. Our encoder works on the image domain. Given a perspective image, we follow the encoder of well-known DCGAN [29] to obtain a 1024-channel 1D code whose length is $20 \times 4^2$. The reason why we choose DCGAN (instead of the other networks) is that DCGAN provides higher reliability in practice. Details are available in the supplementary material. The length of our code equals to the resolution of the icosahedral spherical representation based on two rounds of sub-division. Our decoder works on the sphere domain, and treats our code as the input spherical map. Based on the spherical convolution and up-sampling [20], we alternately extract features (by convolution) and increase the map resolution (by up-sampling). We introduce these operations in the next paragraph. In addition, each spherical convolution is followed by bias adding, batch normalization, and leaky ReLU function. For the one-channel output whose resolution is $20 \times 4^5$, we normalize it as a probability map (see Fig. 1(f)) by Sigmoid function.

We consider a pixel of the icosahedral spherical map to illustrate the spherical convolution and up-sampling. As shown in Fig. 3(b), except for the shape of kernel, spherical convolution is similar to image convolution. By setting the stride as 1, convolution does not change the resolution of sphere. As shown in Fig. 3(c), except for the number of neighbors in the higher-resolution map, spherical up-sampling is similar to image up-sampling.

### 3.3. Loss Function

Our loss function is the combination of three sub-losses. First, we follow [37] to use the pixel-wise mean squared error (MSE) loss. This loss encodes the difference between the predicted and ground truth probability maps. We will introduce how we generate the ground truth map in Section 4. MSE loss is effective in fitting training data. To prevent over-fitting, we exploit the characteristic of probability map for regularization. Specifically, as introduced in Section 3.1, our spherical probability map should exhibit antipodal symmetry. To enforce this constraint, we propose an antipodal symmetry (AS) loss. We define it by the average of the squared differences between all pairs of antipodal pixels on a predicted map. In addition, as shown in Fig. 1(f), many pixels of our probability map should be associated with the probability of 0. To enforce this constraint, we exploit $L_0$ loss [7] to reduce the number of non-zero pixels. Based on the above sub-losses, we define our total loss by
We empirically set the coefficients $\lambda_{\text{MSE}}$, $\lambda_{\text{AS}}$ and $\lambda_1$ as 2, 0.5 and 0.1, respectively. Our experiments demonstrate the effectiveness of all three sub-losses.

4. Generating Ground Truth Probability Map

Given several annotated DDs\(^2\), we aim to generate the ground truth probability map used in Section 3.2. In the field of VP estimation, a method for spherical ground truth map generation does not exist. We first design various baselines and then propose a reliable method.

4.1. Baselines and Their Limitations

**Binary Map** (denoted by Binary). We design this baseline by analogy with [40]. Given several annotated DDs, we generate a spherical binary ground truth map. As shown in Fig. 4(a), we assign 1 to the pixels passed by the annotated DDs, and 0 to the other pixels. However, as will be shown in the experiments, due to too high sparsity of this map, the trained network is inaccurate.

**Unrefined Watson Mixture Model-based Map** (denoted by Unrefined). We design this baseline by analogy with [37]. As shown in Fig. 4(b), we apply Watson distributions [36] with the same isotropic dispersion on pixels passed by the annotated DDs. We choose Watson distribution since its antipodal symmetry is suitable to express DDs (see Section 3.1). This baseline improves the accuracy of the above baseline by reducing the sparsity, as will be shown in the experiments. However, it does consider the noise of annotated DDs, which affects the accuracy. Specifically, on many VP datasets [42, 22, 18], a VP is annotated by computing the intersection of a small number of image lines with the same (manually obtained) labels. It is known that this VP may be unreliable [33, 39], especially when the intersected lines are nearly parallel.

\[
L = \lambda_{\text{MSE}} \cdot L_{\text{MSE}} + \lambda_{\text{AS}} \cdot L_{\text{AS}} + \lambda_1 \cdot L_0.
\]

We empirically set the coefficients $\lambda_{\text{MSE}}$, $\lambda_{\text{AS}}$ and $\lambda_1$ as 2, 0.5 and 0.1, respectively. Our experiments demonstrate the effectiveness of all three sub-losses.

**Refined Watson Mixture Model-based Map** (denoted by Refined). We first follow [33] to alternately refine annotated DDs and update cluster labels of image lines. Then we apply Watson distributions on pixels passed by the refined DDs (see Fig. 4(b)). While this baseline reliably expresses the locations of DDs, it fails to appropriately express the uncertainties of DDs. Specifically, [19] studied the uncertainties of VPs in the image by computing the intersections of all pairs of image lines with the same label. As shown in Fig. 5, the uncertainties of VPs should be expressed by the distributions with various anisotropic dispersions. Similarly, as will be shown in the next section, the uncertainties of DDs should be expressed by the distributions with various anisotropic dispersions. However, Watson distributions used by this baseline exhibit the same isotropic dispersion.

4.2. Map Based on Various Anisotropic Dispersions

To overcome the limitations of the above baselines, we propose a ground truth probability map that appropriately expresses both locations and uncertainties of DDs. First, we follow [33] to alternately refine annotated DDs and update cluster labels of image lines. Then as shown in Fig. 6(a), we map an image line into a great circle on the sphere. As shown in Fig. 6(b), we associate great circles with the updated cluster labels of image lines. As shown in Fig. 6(c), we compute the intersections of all pairs of great circles with the same label. These intersections constitute an antipodally symmetric cluster on the sphere. For all the clusters, their density peaks and various anisotropic dispersions encode the locations and uncertainties of DDs, respectively. Intuitively, a cluster with high-level anisotropic dispersion corresponds to a VP far from the image center.

Based on the intersections computed above, we generate our ground truth probability map. We first define an icosahedral sphere with $20 \times 4^2$ sub-faces (see Section 3.1). Then we compute a frequency histogram of intersections over sub-faces of the icosahedral sphere. Specifically, if an intersection lies within a sub-face, we increase the frequency associated with this sub-face by one. Finally, we normalize the frequencies into $[0, 1]$. As shown in Fig. 6(d), we treat the spherical histogram with normalized frequencies as our ground truth probability map. Our map effectively expresses the distribution pattern of intersections.

\(^2\)Annotated VPs and DDs can be mutually converted (see Fig. 1(e)).
edge of Bingham mixture model. For a point $g$ on the sphere, the probability density function of Bingham distribution is given by

$$B(g | V, k) = \frac{1}{\mathcal{Z}(k)} \exp \left( \sum_{i=1}^{2} k_i (g^\top v_i)^2 \right),$$

(2)

where $v_1$ and $v_2$ are basis vectors, and $V = [v_1, v_2]$; $k_1$ and $k_2$ are concentration parameters, and $k = [k_1, k_2]$. A large magnitude of $k_i$ represents that Eq. (2) highly peaks along $v_i$. To express $M$ Bingham distributions ($M$ represents the unknown-but-sought number of DDs in our context), we use a Bingham mixture model, i.e.,

$$M(g) = \sum_{m=1}^{M} c_m \cdot B(g | V_m, k_m),$$

(3)

where $c_m$ denotes the mixture coefficient of the $m$-th component, and satisfies $c_m > 0$ and $\sum_{m=1}^{M} c_m = 1$.

In the following, we introduce the model fitting. We first follow [38] to sample scatters on the sphere based on the predicted probability map. Intuitively, for a pixel associated with large probability, we sample a large number of scatters within this pixel. Given $N$ sampled scatters $\{g_n\}_{n=1}^{N}$, we aim to cluster them by an unknown-but-sought Bingham mixture model in Eq. (3). To achieve this goal, we maximize a log-likelihood $D$, i.e.,

$$\max_{M, \{V_m, k_m, c_m\}_{m=1}^{M}} \log \prod_{n=1}^{N} M(g_n).$$

(4)

We solve Eq. (4) based on a self-adaptive expectation-maximization algorithm [11]. It can automatically determine the number of DDs $M$. Specifically, we search $M$ in a reasonable range of the number of DDs, e.g., $[1, 6]$. Given a tentative value $i \in [1, 6]$ of $M$, we alternately update the cluster labels of scatters and model parameters. Then we back-substitute the estimated parameters into Eq. (4), obtaining the log-likelihood $D_i$. In addition, we evaluate the complexity of a model with $i$ components by the complexity function $F(i)$ [11]. A small value of $F(i)$ corresponds to low model complexity. By considering both log-likelihood $D_i$ and function $F(i)$, we find the optimal value $i$ based on the minimum message length criterion [35], i.e.,

$$\min_i \left( -D_i + F(i) \right).$$

(5)

This criterion controls the trade-off between fitting quality and model complexity. For algorithm initialization, we assign adjacent scatters with the same cluster labels. Experiments show that our algorithm can robustly converge.

6. Experiments

Datasets. Our experiments are on both real-world [10, 22, 18, 31] and synthetic [41] datasets:
Figure 7. Generality ("G"), accuracy ("A") and efficiency ("E") comparisons on three representative images. "↑", "−" and "↓" represent high, middle and low, respectively. We use image lines to compute $F_1$-score and consistency error, regardless of whether a method requires image lines for VP estimation. In the 3-rd to 7-th columns, a dotted line in the image represents the connection between the midpoint of a clustered image line and an estimated VP. A triplet of numbers below an image represents $F_1$-score, consistency error, and run time. Additional comparisons are available in the supplementary material.

Table 1. Generality and accuracy comparisons on various datasets.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$F_1$-score</td>
<td>Cons. Error</td>
<td>$F_1$-score</td>
<td>Cons. Error</td>
<td>$F_1$-score</td>
</tr>
<tr>
<td>YUD+ [10]</td>
<td>79.55%</td>
<td>0.795 pix.</td>
<td>84.18%</td>
<td>0.682 pix.</td>
<td>78.03%</td>
</tr>
<tr>
<td>VSD [22]</td>
<td>75.36%</td>
<td>0.873 pix.</td>
<td>91.02%</td>
<td>0.769 pix.</td>
<td>70.47%</td>
</tr>
<tr>
<td>NYU-VP [31]</td>
<td>80.20%</td>
<td>0.951 pix.</td>
<td>85.73%</td>
<td>0.782 pix.</td>
<td>76.34%</td>
</tr>
<tr>
<td>SU3 [41]</td>
<td>94.88%</td>
<td>0.782 pix.</td>
<td>96.26%</td>
<td>0.598 pix.</td>
<td>94.37%</td>
</tr>
</tbody>
</table>

- YUD+ [10, 18] consists of 102 outdoor and indoor images with 3~6 vertical, horizontal and/or sloping VPs.
- VSD [22] consists of 97 outdoor images with 4~6 vertical and horizontal VPs.
- NYU-VP [31, 18] consists of 1449 indoor images with 1~6 vertical, horizontal and/or sloping VPs.
- SU3 dataset [41] consists of 23,000 outdoor images with 3~4 vertical, horizontal and/or sloping VPs.

We extract image lines by LSD [34] to estimate VPs and/or evaluate accuracy. For deep learning-based methods, we treat 80% and 20% of images of each dataset as training and testing images, respectively. We follow [26] to combine all the training images to train a single network. Then we test this network on testing images of each dataset independently. For traditional methods, we test it using the above testing images of each dataset independently. Additional information is available in the supplementary material.

Evaluation Criteria. In our context, high generality represents that a method can detect various VPs, e.g., non-orthogonal horizontal VPs and sloping VPs (see Fig. 7). We follow [22, 21] to evaluate generality by $F_1$-score that considers both precision and recall of image line clustering. In addition, for accuracy evaluation, we choose the widely-used consistency error [33, 39, 25, 22]. Consistency error in the image is more reasonable than angle evaluation in 3D [41] since uncertainty originates from the image [39]. Specifically, an estimated VP and midpoint of an image line $l$ associated with this VP define a virtual line $v$. Consistency error represents the distance from an endpoint of the line $l$ to the virtual line $v$. Additional illustrations are available in the supplementary material. On multiple images of a dataset, we report the mean of each metric.

Implementation Details. We use Adam [17] to minimize our loss. Our learning rate is $10^{-4}$, batch size is 16, and number of epochs is 30. We implement our method with TensorFlow and conduct tests on a computer equipped with
TITAN Xp GPU and Xeon E5-2680 v4 CPU.

6.1. Comparison with State-of-the-art Approaches

We denote our deep learning-based method that does not rely on image lines and also can automatically determine the number of DDs by DL-nL-auto. We compare it with the state-of-the-art approaches introduced in Section 2:

- The traditional method [25] relies on image lines and also assumes 3 mutually orthogonal VPs. We denote it by TR-L-3.
- The traditional method [22] relies on image lines and can automatically determine the number of DDs. We denote it by TR-L-auto.
- The deep learning-based method [41] does not rely on image lines and also requires prior knowledge of the number of VPs. It assumes 3 VPs when prior knowledge is unavailable. We denote it by DL-nL-3.
- The deep learning-based method [18] relies on image lines and can automatically determine the number of DDs. We denote it by DL-L-auto.

Generality and Accuracy. As shown in Fig. 7 and Table 1, TR-L-3 only works well in Manhattan world. On the images with non-orthogonal and sloping VPs, it leads to unsatisfactory recall, which affects the $F_1$-score. TR-L-auto fails to handle sloping VPs and also is prone to neglecting some VPs associated with a small number of image lines. DL-nL-3 can find three non-orthogonal VPs, and thus is more general than TR-L-3. However, it still fails to avoid under- or over-detection of VPs due to the assumption of three VPs. DL-L-auto can hardly detect VPs associated with a small number of image lines due to high difficulty of valid sampling. In contrast, our DL-nL-auto can predict a reliable spherical probability map and detect all the DDs based on this map. In addition, traditional methods lead to smaller consistency error than deep learning-based approaches thanks to geometric constraints. Our DL-nL-auto is the most accurate deep learning-based method.

Efficiency. As shown in Fig. 7 and Table 2, TR-L-3 is not very efficient in non-Manhattan worlds. It treats image lines associated with non-orthogonal VPs as outliers and thus leads to numerous iterations. TR-L-auto is time-consuming due to high-dimensional parameter space and highly non-linear cost function. The efficiency of DL-nL-3 is moderate due to the simplification of its coarse-to-fine inference strategy. DL-L-auto provides unsatisfactory efficiency for a relatively large number of VPs. Its time cost is mainly caused by sequential computation of sampling weights. Our DL-nL-auto is relatively efficient thanks to concise network and moderate resolution of map. In addition, compared with image line-based methods, deep learning-based approaches lead to smaller variances of runtime.

6.2. Ablation Studies

Spherical Expression. We express our probability map based on icosahedral spherical representation (see Section 3.1). We compare our network based on this representation with a baseline based on the equi-angular discretization. We introduce the architecture of baseline in the supplementary material. For a fair comparison, we set the resolution of equi-angular discretization as 200×100=20,000 pixels (recall that the icosahedral spherical representation consists of $20 \times 4^5 = 20,480$ pixels). As shown in Table 3 and Fig. 8, icosahedral spherical representation is more accurate than equi-angular discretization. The reason is that this representation provides uniform pixel distribution, and
thus facilitates estimating arbitrary orientations of DDs.

In addition, we express the probability map on the sphere instead of hemisphere (see Section 3.1). We use the ground truth maps expressed by sphere and hemisphere to train our network, respectively. Table 4 and Fig. 9 show that sphere leads to higher accuracy. The reason is that sphere can keep the integrity of probability distribution. In contrast, when generating hemisphere, a dividing line of sphere may split a probability distribution. Accordingly, hemisphere results in unreliable probability prediction around the dividing line.

**Ground Truth Probability Map.** We design baselines Binary, Unrefined and Refined and our method (see Section 4). We use different ground truth maps generated by these methods to train our network. As shown in Fig. 10, Binary leads to low accuracy since too high sparsity of ground truth map affects network training. The accuracy of Unrefined is unsatisfactory since it neglects the noise of annotated DDs. Refined improves the accuracy to some extent. However, due to inappropriate expression of uncertainties of DDs, it can hardly handle VPs far from the image center. Our method provides the highest accuracy since it reasonably expresses both locations and uncertainties of DDs.

**Loss Function.** Our loss function is the combination of MSE, AS, and \( L_0 \) sub-losses (see Section 3.3). We test our network trained by various combinations of sub-losses. Additional tests on coefficient variation are available in the supplementary material. As shown in Table 5 and Fig. 11, MSE loss is effective in fitting the probability maps, but the accuracy is limited by slight asymmetry and too wide dispersions of distributions. AS and \( L_0 \) losses both improve the accuracy. The reason is that they can align the peaks of distributions and compress small non-zero probabilities respectively, and thus effectively prevents over-fitting.

**DD Detection.** We detect DDs by fitting a Bingham mixture model instead of non-maximum suppression (see Section 5). Given the same predicted probability map, we compare these strategies. As shown in Fig. 12, non-maximum suppression is prone to resulting in under- or over-detection especially when two DDs are relatively close. In contrast, our method is robust since it is free of threshold.

7. **Conclusions**

The proposed VP estimation method is independent of image lines and also can automatically determine the number of VPs, providing high generality. Moreover, it achieves satisfactory accuracy and high efficiency thanks to novel spherical representation, loss function, ground truth map generation, and DD detection. Therefore, it is more practical than existing methods failing to simultaneously guarantee generality, accuracy, and efficiency.

**Acknowledgments.** Yun-Hui Liu was supported by the Hong Kong Centre for Robotics, RGC via grant 14207320, the CUHK VC Discretionary Fund, and Shenzhen Municipal Government via the Shenzhen-HK Collaboration Zone Project. **Kyunghoon Joo** was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-01336, Artificial Intelligence Graduate School Program (UNIST)) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2021R1C1C1005723). **Pyojin Kim** was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1061397). **Kuk-Jin Yoon** was supported by Institute of Information and Communications Technology Planning & Evaluation (IITP) Grant funded by Korea Government (MSIT) (No. 2020-0-00440, Development of Artificial Intelligence Technology that Continuously Improves Itself as the Situation Changes in the Real World).
References


[38] Menghua Zhai, Scott Workman, and Nathan Jacobs. Detecting vanishing points using global image context in a Non-Manhattan world. In CVPR, 2016. 1, 2, 5

[39] Lilian Zhang, Huimin Lu, Xiaoping Hu, and Reinhard Koch. Vanishing point estimation and line classification in a Manhattan world with a unifying camera model. IJCV, 2016. 1, 2, 4, 6

[40] Ziheng Zhang, Zhengxin Li, Ning Bi, Jia Zheng, Jinlei Wang, Kun Huang, Weixin Luo, Yanyu Xu, and Shenghua Gao. PPGNet: Learning point-pair graph for line segment detection. In CVPR, 2019. 4
